Layered Scene Models from Single Hazy Images

Lingyun Zhao, Miles Hansard, Andrea Cavallaro

Abstract—This paper describes the construction of a layered scene model, based on a single hazy image that has sufficient depth variation. A depth map and radiance image are estimated by standard dehazing methods. The radiance image is then segmented into a small number of clusters, and a corresponding scene plane is estimated for each. This provides the basic structure of a layered scene model, without the need for multiple views, or image correspondences. We show that problems of gap filling and depth blending can be addressed systematically, with respect to the layered depth structure. The final models, which resemble cardboard 'pop-ups', are visually convincing. An implementation is described, and subjective depth preferences are tested in a psychophysical experiment.

Index Terms—Scene modelling, single view reconstruction, image dehazing.

1 Introduction

3D scene reconstruction has many applications in multimedia content generation, outdoor navigation, cultural heritage and virtual environments. In recent years, free viewpoint television (FTV) and virtual reality (VR) technologies, which allow users to control the viewpoint, have begun to appear. Content production, however, remains a relatively complicated and time consuming process, which often requires special equipment, a large number of photographs, manual interaction, or some combination of these. This makes it difficult for naive users to create an interactive 3D model.

This paper describes a system that can semi-automatically create a 'layered' 3D model, from a single outdoor image, taken in hazy conditions. The layered scene model was first proposed by Wang et al. [37], who used layers to represent a scene with moving objects, via a motion segmentation algorithm. Rather than using multiple views, range scanners, or complex scene models, we use the depth information provided by atmospheric effects. Moreover, we use the dehazed radiance image to provide the colour information for compositing. We estimate the depth of each layer from the local haze content of the images. The raw point cloud 3D model, produced by back projecting the estimated depth map, may be unsuitable for general users to display. Hence, we create a visually compelling layer-based 3D scene, which can be displayed via the standard graphics pipeline. The foreground objects are automatically pasted onto different planes, with appropriate positions and orientations, and without gaps at the plane boundaries when the viewpoint is varied. Our plane-based approach is similar to the creation of a cardboard 'pop-up' illustration in a children's book, as shown in Fig. 1. A possible application scenario is that the hazy outdoor photos would be processed rapidly, so that users would be able to interact immediately with the resulting models (e.g. via HTML5 on mobile devices). Another potential application is that with FTV or VR devices, users would also generate their own lifelike experience, for example, in simulations for house construction or scenery viewing. Our approach could also make VR game creation less time consuming.

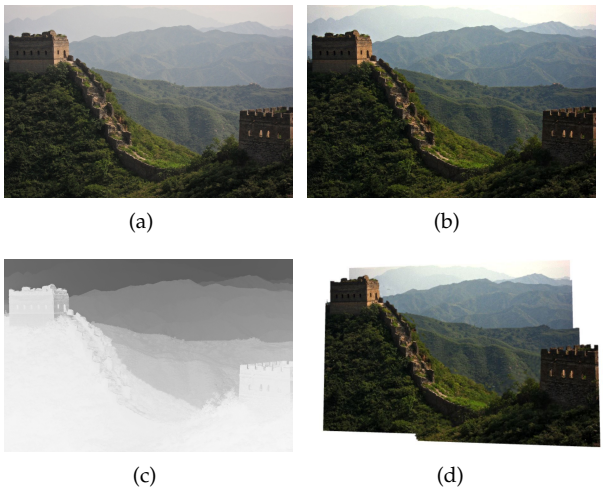


Fig. 1: Our system constructs a plane-based 3D model from a single hazy image, without losing any information from the original image. (a) The original image. (b) The resulting radiance image $\mathcal{L}_0^{\text{rgb}}(x, y)$ from dehazing. (c) The by-product from dehazing, which is the range map $r(x, y)$. We make use of the depth information to construct the 3D scene. (d) A dehazed novel view from our 3D model.

The challenges of generating a pop-up image are as follows. Firstly, the original image does not contain explicit depth information. In order to get the depth information, we usually need multiple views, or some other source of information. This can be difficult to obtain, especially if the image quality is poor (e.g. due to haze), or if computational resources are limited (e.g. on mobile devices). Secondly, even if depth can be estimated, it is important to resolve visibility correctly, in order to produce a convincing model. This usually requires a surface model of some kind, such as a mesh, which can again be difficult to estimate. Thirdly, the inherent problem of pop-up modelling from single view im-

• *L. Zhao, M. Hansard and A. Cavallaro are with the Centre for Intelligent Sensing, Queen Mary University of London, Mile End Road, London E1 4NS, United Kingdom.*
E-mail: lingyun.zhao, miles.hansard, a.cavallaro@qmul.ac.uk.

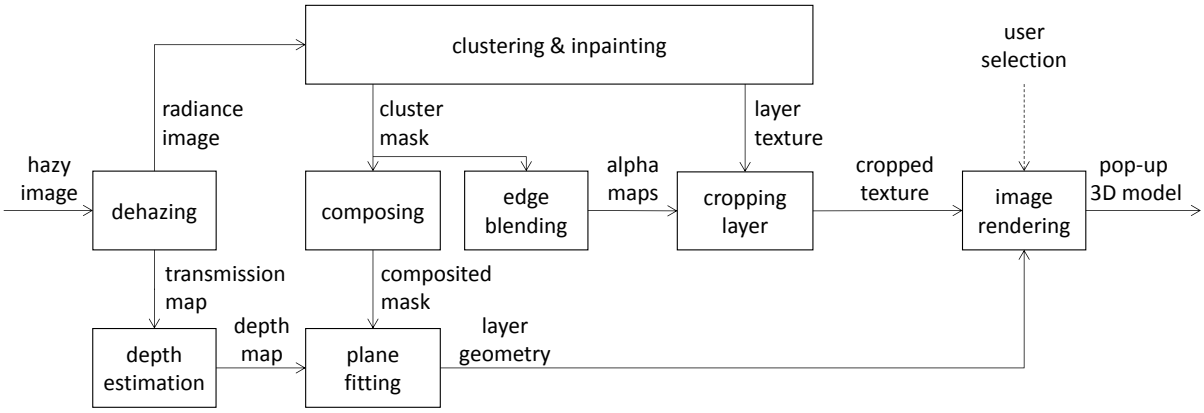


Fig. 2: Block diagram of the proposed system for creating a pop-up 3D model. The radiance image $\mathcal{L}_0^{\text{rgb}}(x,y)$ and transmission map $t(x,y)$ are obtained from a single hazy image by dehazing process. The clustering & inpainting output all cluster masks $m_{1:N}$ and layers' textures $w_{1:N}$. The total number of clusters N is user defined. The layer texture $\mathcal{L}_{1:N}^{\text{rgb}}(x,y)$ is obtained by matting the alpha channel $\mathcal{L}_{1:N}^{\alpha}(x,y)$ and inpainting results $w_{1:N}$. A composited mask map M is obtained by composing all object masks. M is an input to plane fitting, together with the estimated depth map $\mathcal{L}_0^{\text{z}}(x,y)$. The output of plane fitting is the depth for each plane $\mathcal{L}_{1:N}^{z}(x,y)$. All of the depth and colour information is transferred to image rendering to build a layered scene model. Finally, users can interactively control the resulting depth structure.

TABLE 1: Table of notations

$t(x,y)$	transmission map
$r(x,y)$	range map
i	cluster/layer number
m_i	cluster mask
m_i'	marker image
w_i	inpainted image
N	number of clusters
M	mask map
$\mathcal{L}_0^{\text{rgb}}(x,y)$, m_0'	radiance image
$\mathcal{L}_0^{\text{z}}(x,y)$	depth map
$\mathcal{L}_i^{z}(x,y)$	layer's depth map
$\mathcal{L}_i^{\alpha}(x,y)$	layer's alpha channel
$\mathcal{L}_i^{\text{rgb}}(x,y)$	layer's alpha premultiplied texture

ages is the gaps between different layers; these are occluded regions in the original view, which are revealed in the virtual view as a result of a new viewpoint. These holes can be sizeable and, in the single view setting, they cannot be resolved from alternative source images. It follows that the holes must be 'inpainted', which poses several challenges.

The main contribution of this paper, which builds on our earlier work [39], is to offer a sophisticated solution to the problem of scene reconstruction based on layer modelling for a real world hazy single image. Note that the scene need not be noticeably foggy in order to provide depth information. Ordinary atmospheric scattering is sufficient, provided that the scene contains some distant regions. Our contribution includes a new geometric parameterization of pop-up 3D modelling. In addition, an appropriate image blending model is introduced for depth boundaries in the synthetic model. Our proposed method allows the user to define the number of object segments and to cluster them by graph-cut. We fill the gaps that appear when the viewpoint is changed by applying a texture synthesis inpainting algorithm. We implemented our proposed rendering method in HTML5, which can be displayed interactively in any browser or WEBKIT based program, for example. The complete pipeline is shown in Fig. 2 and some commonly used notations are listed in Table 1.

We offer solutions to some of the problems of previous approaches [39] to layer-based scene reconstruction from a single image:

- A novel inpainting scheme is introduced to improve the quality and efficiency by applying clustering algorithms in an appropriate way.
- Depth estimation is applied to the range map obtained from dehazing, in order to improve the accuracy of the estimated layers.
- A complete implementation is described to visualize our model.
- The perceived quality of the resulting models is evaluated in a perceptual experiment.

This paper is organized as follows. Section 2 discusses the state-of-the-art in layered scene representations. The geometry of our pop-up model is developed in Section 3. In Section 4 we introduce a pipeline to create a haze-based pop-up model, including a discussion of the dehazing process. Section 5 provides the details of the HTML5 pop-up rendering, and includes some qualitative comparisons with related scene models. In Section 6, we discuss two subjective experiments based on hazy outdoor images and ground-truth images on depth structure preferences with user interactive control. Finally, Section 7 concludes the paper, and outlines future work.

2 RELATED WORK

In this section, we focus on methods that produce visually pleasing scene models based on a single view image, without necessarily being veridical or complete. According to the geometry of the scene model, we classify the existing

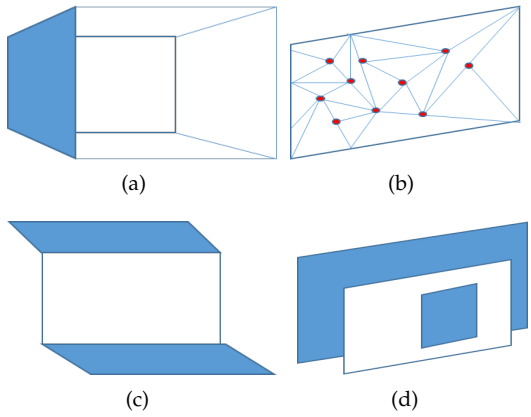


Fig. 3: Schematic single view scene models. (a) Boxed model: Horry et al. [15]. (b) Meshed model with user guide: Zhang et al. [38]. (c) Staggered model: Hoiem et al. [14] and Saxena et al. [28]. (d) Layered model: Swirski et al. [34] and our proposed method.

methods as boxed, meshed, staggered or layered (see Fig. 3).

Depending on the need for user-specified constraints to obtain depth information, we classify state of the art methods as interactive or automatic. *Interactive methods* perform user guided modelling. Liebowitz et al. [19] and Criminisi et al. [4] developed an accurate architectural 3D modelling system, which used one or two views. Additional constraints were obtained by analysing perspective distortions of regular structures, in order to perform metric rectification of the images. This system required considerable user guidance, to specify a set of parallel lines, length ratios, and orthogonality relationships. Horry et al. [15] modelled the scene as a frontally open box, which has a bottom plane, a top plane, a background plane and two side planes (see Fig. 3(a)). Users are required to specify the coordinates of their box model, including its vanishing point, and then to label foreground objects manually, and assign them to planes. Although this method achieved an impressive walk in effect, the limitation is that the scene must be amenable to the box style model. In our work, the positions and orientations of the planes are freely determined by the scene structure, rather than being limited to a predefined arrangement. Similarly, Zhang et al. [38] modelled general scenes by requiring users to specify geometric constraints [12], such as normal directions. Based on these constraints, they optimized for the best 3D model (Fig. 3 (b)). Most recently, Swirski et al. [34] proposed another layered model (Fig. 3 (d)), which requires depth information to be provided in advance. Their results still have minor rubber sheet effects at the occluding contours of foreground objects, resulting from depth interpolation. As Swirski et al. [34] used external input from a Kinect device, their method is classified with the *interactive* ones.

Instead of requiring user specified constraints, some work introduced an *automatic method* to construct a planar 3D model. Hoiem et al. [14] proposed a method that learns a statistical model of scene structure. The major assumption is that the scene is composed of a single ground plane, a

TABLE 2: Approaches to single view 3D scene modelling. *Automatic* methods do not need user-specified geometry constraints or other special equipments to obtain depth information.

	automatic	depth estimation	semantic interpretation	soft segmentation	free-form scene	interactive adjustment
Liebowitz [19]			✓			
Criminisi [4]			✓			
Horry [15]		✓		✓		✓
Zhang [38]					✓	
Hoiem [14]	✓					
Saxena [28]	✓	✓				
Swirski [34]			✓	✓	✓	
Karsch [17]	✓		✓		✓	✓
Proposed	✓	✓	✓	✓	✓	✓

ceiling plane and multiple planar objects standing on the ground at right angles (Fig. 3 (c)). Under this assumption, they labelled each pixel as belonging to one of three clusters, namely ground, vertical or sky. Then a simple, scaled 3D model is constructed by estimating the horizon position of each cluster. However, this model cannot accommodate more complicated scene arrangements, such as overlapping buildings and people. Karsch et al. [17] introduced a way to composite a synthetic object into a real scene image. They recovered scene geometry from a single image, which improved methods introduced by Hoiem et al. [14] by incorporating geometric constraints. Our system has no geometric constraints: instead, it gets depth information from a hazy photograph of the scene.

Saxena et al. [27, 28] presented an algorithm for estimating depth information from the features in a monocular image. In many cases, however, their depth maps are not accurate enough to produce visually pleasing 3D models. Their method also relies on good quality of image features, which therefore makes hazy images unsuitable. More recent work proposed by Saxena et al. [29] used a Markov Random Field (MRF) to infer a set of plane parameters, which express both the 3D orientation and position of each surface patch. They also incorporated object recognition information into the MRF to improve the results. However, the resulting models may be visually incomplete.

Relevant previous works discussed in this section are summarised and compared in Table 2. *Interactive adjustment* methods can allow users to modify the 3D structure.

3 Pop-up model

This section gives a formal definition of the layered scene model, and introduces a new geometric parameterization of the pop-up scheme.

3.1 Scene representation

We represent each layer as a plane, with colours and depths that are indexed by the image coordinates (x, y), as follows:

$$\mathcal{L}(x,y) = \Big(\mathcal{L}^{\text{rgb}}(x,y),\ \mathcal{L}^{\alpha}(x,y),\ \mathcal{L}^{z}(x,y)\Big). \qquad (1)$$

Here $\mathcal{L}^{\text{rgb}}(x,y)$ is a layer texture image, premultiplied by the corresponding alpha channel $\mathcal{L}^{\alpha}(x,y)$, and $\mathcal{L}^{z}(x,y)$ is a depth map. Recall that premultiplied foreground and background colours $\mathcal{F}^{\text{rgb}}$ and $\mathcal{B}^{\text{rgb}}$ can be combined by the standard image compositing 'over' operation $\mathcal{F}^{\text{rgb}} \odot \mathcal{B}^{\text{rgb}} = \mathcal{F}^{\text{rgb}} + (1 - \mathcal{F}^{\alpha})\mathcal{B}^{\text{rgb}}$ [25], where the opacity of each foreground pixel is determined by the corresponding alpha map, $\mathcal{F}^{\alpha}$. Note that for $\mathcal{F}^{\alpha} = 1$ (opaque foreground), the foreground is returned, whereas for $\mathcal{F}^{\alpha} = 0$ (transparent foreground), the background is returned. We require a more general operation $\mathcal{L} = \mathcal{L}_i \odot \mathcal{L}_j$, as used in the stereo-based model of Szeliski and Golland [35], which we apply to the layers:

$$\begin{aligned}\mathcal{L}(x,y) &= \bigodot_{j=1}^{N} \mathcal{L}_{i_j}(x,y) \\ &= \mathcal{L}_{i_1} \odot \mathcal{L}_{i_2} \cdots \odot \cdots \mathcal{L}_{i_N}.\end{aligned} \tag{2}$$

Here the operator $\bigodot_j$ composites each layer $\mathcal{L}_k$, with premultiplied alphas, over the current background layer. The composition is performed in depth order, from $i_1 = \arg\min_k \mathcal{L}_k^z$ to $i_N = \arg\max_k \mathcal{L}_k^z$. In principle, the depth order could vary from pixel to pixel but, in practice, contiguous pixels will be composited in the same way. Note that we may interpret the 'composited depth map' as the minimum depth $\mathcal{L}_{i_1}^z$ at each pixel (x,y).

Note that our method does not model the relief structure of individual objects, as in outdoor scenes the depth variation due to surface shape is in general relatively small in comparison to the distances between different surfaces. The layered model is therefore appropriate, because the dehazing process already assumes an outdoor scene.

3.2 Scene geometry

In this section, we introduce a parameterization of the scene geometry, for pop-up modelling, which is designed to satisfy the following requirements:

Efficiency: The parameterization should preserve *flatness* in 3D, so that standard clipping and texture mapping operations can be used.

Interactivity: It should be possible to vary the depth structure of the pop-up, in an intuitive way.

Consistency: It should be possible to obtain the original image, as a projection of the scene model, given the appropriate parameter settings.

The pop-up geometry described in this section is closely related to the standard *viewing frustum* in computer graphics. The virtual camera is located at the apex of the frustum, while the near and far planes are located at viewing distances Z_{near} and Z_{far} respectively, as shown in Fig. 4. The rectangular boundaries of the view frustum in the near and far planes can be formulated as a function of the viewing distance, the horizontal field of view ϕ, and the aspect ratio of the image w/h, where w and h are image width and height respectively:

$$\begin{aligned}H_{\text{near}} &= 2\tan(\phi/2)\, Z_{\text{near}} \\ W_{\text{near}} &= (w/h)\, H_{\text{near}}.\end{aligned} \tag{3}$$

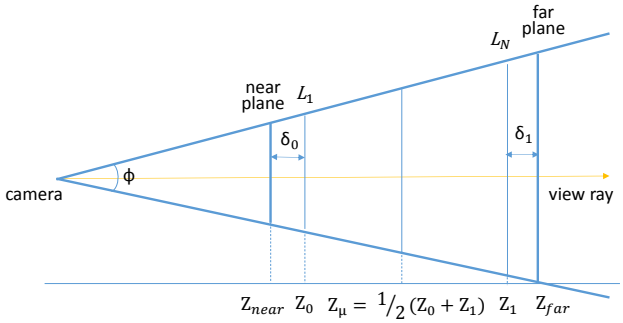


Fig. 4: Definition of the viewing frustum. The region between the near and far clipping planes is rendered in the image.

The far width W_{far} and height H_{far} are obtained by replacing Z_{near} with Z_{far} in equation (3). We make sure that all of the layers are fitted into the camera viewing frustum so that the hard edges due to different sizes of each layer are not visible. In addition, the 3D experience is determined entirely by the parallax (scene difference caused by different viewpoints), induced by moving the camera in a direction orthogonal to the axis of the frustum.

The input to the pop-up system consists of the radiance image $\mathcal{L}_0^{\text{rgb}}(x,y)$ and associated depth map $\mathcal{L}_0^z(x,y)$, obtained from the dehazing process (Fig. 2). The image coordinates (x,y) are related to the corresponding scene coordinates (X,Y,Z) by perspective projection, using the standard pinhole camera model:

$$x = f\,X/Z \quad \text{and} \quad y = f\,Y/Z, \tag{4}$$

where Z is the *true* depth of the scene point $\mathbf{P}$. If the image has unit width, then the focal length is $f = \frac{1}{\tan(\phi/2)}$, where ϕ is the horizontal field of view. These relations can be expressed in homogeneous coordinates as

$$\mathbf{P} \simeq \big(X, Y, Z, 1\big)^{\top} \simeq \big(x/f,\ y/f,\ 1,\ 1/Z\big)^{\top}, \tag{5}$$

where '$\simeq$' means equality up to an overall scale factor. Equation (5) could be used to generate a pop-up, e.g. by setting $f = 1$ and $Z = z$. However, z will generally be a distorted estimate of Z, owing to assumptions and unknown parameters in the dehazing process.

To achieve a visually pleasing result, the depth distortions could be interactively adjusted by defining a more general back projection function

$$\mathbf{P} \leftarrow \big(X(x,y,z),\ Y(x,y,z),\ Z(x,y,z)\big)^{\top}.$$

As previously indicated, in practice there are strong constraints on the three coordinate functions. Firstly, the global depth distortion is likely to preserve the depth order of the scene, so the mapping $z \to Z$ should be monotonic. Secondly, the scene layers are rendered as textures, so it is highly desirable that planar layers remain planar. This means that each layer can be represented by two triangles, which permits very fast rendering. The most general functions that preserve flatness are projective *homographies*, as illustrated in Fig. 5. Mathematically, this pop-up model is

$$\mathbf{P} \leftarrow \mathbf{H}\mathbf{p} \quad \text{with} \quad \mathbf{p} = \big(x,\ y,\ 1,\ 1/z\big)^{\top}, \tag{6}$$

(a) (b) (c)

Fig. 5: Schematic 2D illustration of the parameterization, with the viewpoint at the apex (bottom) of the frustum. (a) Equal steps in the input depth distribution. (b) Projective transformation of the Z coordinate, as used in this work. (c) General projective transformation of X and Z coordinates. The 3D cases are analogous, with lines replaced by planes.

where $\mathbf{H}$ is a 4×4 matrix and the input point $\mathbf{p}$ has been defined by analogy with (5). Note that an *orthographic* pop-up could be created by using the alternative format $\mathbf{p}_{\text{orth}} = (x, y, z, 1)^\top$ for the input point in (6).

3.2.1 Depth mapping

The homography matrix $\mathbf{H}$ in (6) contains 16 numbers, which are defined up to an overall scale. It is clearly not practical to set all of these interactively. Hence the form of the matrix will be restricted to the block diagonal

$$\mathbf{H} \simeq \begin{pmatrix} h\,\mathbf{I}_{2\times 2} & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & \mathbf{H}_z \end{pmatrix} \quad \text{with} \quad \mathbf{H}_z \simeq \begin{pmatrix} a & b \\ c & 1 \end{pmatrix}, \tag{7}$$

where the parameters (a, b, c) control the depth transformation. The parameter h acts as an inverse focal length, which maps pixel units into the scene space. Combining the above definitions with the pop-up model (6) gives the 3D point $\mathbf{P} = (X, Y, Z, 1)^\top$, with

$$X \leftarrow hZ'x \quad Y \leftarrow hZ'y \quad \text{and} \quad Z \leftarrow aZ' + \frac{b}{cz+1} \\ \text{where} \quad Z' = \frac{z}{cz+1}. \tag{8}$$

Note that if $(a, b, c) = (1, 0, 0)$, and $h = 1/f$, then these relations are equivalent to (5). More generally, the pop-up depth Z is related to the haze-based estimate z by a 1D homography:

$$Z = \frac{az+b}{cz+1} \simeq \mathbf{H}_z \begin{pmatrix} z \\ 1 \end{pmatrix} \tag{9}$$

with $\det(\mathbf{H}_z) \neq 0$. If $a \geq 1$ and $c \neq 0$, then a 'compressive' transformation is performed on the dehazing depth z. The model described above ensures that the depth planes stay *parallel*, as well as *flat*; this leads to a simple interactive control procedure, as described below[1].

3.2.2 Scene projection

From an abstract point of view, the projection of the pop-up (6) is determined by a 4×4 camera matrix $\mathbf{G}$, so that

$$\mathbf{p}' \simeq \mathbf{GHp}, \tag{10}$$

1. It would be straightforward to allow 'depth skew', if necessary, by incorporating further parameters in the matrix $\mathbf{H}$ in (7).

where $\mathbf{p} = (x, y, 1, 1/z)^\top$, as in (6), and the perspective division is by the *third* coordinate of $\mathbf{GHp}$. The matrix $\mathbf{G}$ is the product of a focal length matrix $\text{diag}(f, f, 1, 1)$, and a rigid 3D motion $(\mathbf{R}, \mathbf{t})$. Retaining the fourth row $(0, 0, 0, 1)$ of the projection matrix $\mathbf{G}$ means that the inverse depth is recovered in the last coordinate of $\mathbf{p}'$. It also means that the transformation (10) is invertible, and hence permits an equivalent interpretation, in terms of scene planes:

$$\boldsymbol{\pi}' \simeq \mathbf{G}^{-\top}\mathbf{H}^{-\top}\boldsymbol{\pi},$$

where $\boldsymbol{\pi} \simeq (\mathbf{n}, -d)^\top$, with surface normal $\mathbf{n}$ and perpendicular distance d from the origin. However, for implementation purposes it is more convenient to represent the planes by the corner points of their bounding boxes, as described in Sec. 5.1.

It remains to be shown that the original image geometry can be exactly recovered from the above model, under appropriate conditions, and that this process is compatible with the standard graphics pipeline. Firstly, note that the inverse of the pop-up matrix (7) can be factored as

$$\mathbf{H}^{-1} \simeq \mathbf{S}\,\mathbf{T}, \tag{11}$$

where $\mathbf{T}$ is a pure translation. In particular, the pop-up of $\mathbf{p}$ as $z \to 0$ in (9) is the 3D point $(0, 0, b)^\top$, which leads us to set

$$\mathbf{T} = \begin{pmatrix} \mathbf{I}_{3\times 3} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix} \quad \text{with} \quad \mathbf{t} = (0, 0, -b)^\top. \tag{12}$$

We now divide the left hand side of (11) by $\mathbf{H}_{33}^{-1}$, to absorb the projective scaling, and solve for $\mathbf{S}$. This results in

$$\mathbf{S} = \begin{pmatrix} s\,\mathbf{I}_{2\times 2} & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & \mathbf{S}_Z \end{pmatrix} \quad \text{with} \quad \mathbf{S}_Z = \begin{pmatrix} 1 & 0 \\ -c & s\,h \end{pmatrix} \\ \text{and} \quad s = \frac{\det(\mathbf{H}_z)}{h}. \tag{13}$$

If the pure translation $\mathbf{T}$ is applied to the scene, as a standard model view matrix, then $\mathbf{S}$ can be interpreted as a camera. In particular, the focal length s in (13) can be obtained geometrically, by considering the pop-up of a notional point $(1, 0, 1, 1)^\top$ at one edge of the image:

$$s \equiv \frac{1}{\tan(\phi/2)} = \frac{Z-b}{X} = \frac{a-bc}{1+c} \Big/ \frac{h}{1+c},$$

where ϕ is the field of view defined by $\mathbf{S}$. This is consistent with the algebraic definition of s in (13), where $\det(\mathbf{H}_z) = a - bc$, as can be seen from (7). Hence $\mathbf{ST}$ is the exact 'pop-down' of $\mathbf{P}$ in (6); the original image is obtained after a Z-translation and change of focal length. Furthermore, the matrix $\mathbf{S}$ can postmultiply the standard 4×4 camera matrix, without changing its form (i.e. the pattern of zero and nonzero entries); in particular, the clipping transformation operates as normal.

3.2.3 Interactive control

The three parameters (a, b, c) in (7) are determined by a combination of fixed scene constraints, and interactive user control. The first constraint is to map the estimated depths into the given frustum. Let the near and far clipping planes be located at Z_{near} and Z_{far}, respectively, as in Fig. 4. Now

define small positive constants (δ_0, δ_1), so that the following limits can be set to make sure that the pop-up scene is fully located inside the field of view:

$$Z_0 = (1+\delta_0)\, Z_{\text{near}} \quad \text{and} \quad Z_1 = (1-\delta_1)\, Z_{\text{far}}. \tag{14}$$

The transformation (14) will be required to perform a mapping $z_{\min} \to Z_0$ and $z_{\max} \to Z_1$, where $z_{\min}$ and $z_{\max}$ are the empirical minimum and maximum values of $\mathcal{L}_i^z(x,y)$, over all layers. This leaves one free degree of freedom to be determined, which can be done by requiring $z_\mu \to Z_\lambda$, where z_μ is a fixed intermediate depth, in the haze data. One possible choice is to define z_μ in relation to the limits of the data:

$$z_\mu = \tfrac{1}{2}\big(z_{\min} + z_{\max}\big). \tag{15}$$

Alternatively, a more scene dependent definition would be the median depth. Meanwhile, the single parameter Z_λ is set by the user, via setting a parameter $\lambda \in [0,1]$, such that

$$Z_\lambda = (1-\lambda) Z_0 \,+\, \lambda Z_1. \tag{16}$$

The requirements $z_{\min} \to Z_0$, $z_\mu \to Z_\lambda$, $z_{\max} \to Z_1$ will now be used to determined the three unknowns (a,b,c) in equation (7).

The matrix $\mathbf{H}_z$ in equation (6) can be computed by a variant of the 'Direct Linear Transformation' algorithm [33]. Let bold face letters represent the homogenous coordinates of the z-values, e.g. $\mathbf{z} \simeq (z,1)^\top$ and $\mathbf{Z} \simeq (Z,1)^\top$. Also define the 'perp' notation $(x,y)^\perp = (-y,x)$. A pair of 2D vectors $\mathbf{u}$ and $\mathbf{v}$ are *collinear* (scalar multiples) if $\mathbf{u}^\perp \cdot \mathbf{v} = 0$. This is a suitable definition of 'equality', in this context, because it does not depend on the unknown homogeneous scale factors. The three geometric constraints, defined above, can now be expressed algebraically as

$$\mathbf{Z}_0^\perp \cdot \mathbf{H}_z\, \mathbf{z}_{\min} = 0, \quad \mathbf{Z}_\lambda^\perp \cdot \mathbf{H}_z\, \mathbf{z}_\mu = 0, \quad \mathbf{Z}_1^\perp \cdot \mathbf{H}_z\, \mathbf{z}_{\max} = 0.$$

This linear system can be solved in closed form, which means that the parameters (a,b,c) can be determined at interactive rates, e.g. if λ is set via mouse wheel control. Furthermore, the above procedure can easily be generalized to the case of more than three (possibly inconsistent) depth constraints [12].

4 Pop-up construction

This section introduces the method we propose to build a pop-up model based on a single hazy image.

4.1 Dehazing

The difficulty of solving the haze removal problem comes from the joint uncertainty of scene radiance and scene transmission. To deal with this ill-posed problem, some previous research estimated depth maps either from multiple images or a single image with user interaction or special devices [22, 23, 24].

In particular, He et al. [13] introduced the Dark Channel Prior (DCP) to infer the scene transmission. They argue that, in the absence of atmospheric effects, a typical image patch should have one 'dark' RGB channel in true radiance. The assumption here is that a local image patch will contain either dark or colourful objects (or both). Under this assumption, the lowest radiance in a haze free patch would be dominated by the addition of atmospheric light in the observed image. The denser the haze, the higher the intensity in the dark channel. An atmospheric transmission map could therefore be estimated from the observed intensity of the dark channel.

Han and Wan [11] proposed a fast dark channel depth map approximation method for dehazing. They also refined the transmission map, but only for the edge areas, where the depth changes dramatically and is not uniform in a local patch. Matlin et al. [20] combined the dark channel approach with non-parametric denoising. Instead of assuming zero minimal values as in the traditional approach, Gibson and Nguyen [9, 10] used a minimum volume ellipsoid approximation to improve the dark channel prior [13], by finding the darkest pixel average inside each ellipsoid. Meng et al. [21] introduced contextual regularisation and filter each colour channel by a minimum filter with a moving window. However, this method shares the problem with He et al. [13], in that the estimated transmissions are not accurate enough to deal with bright objects or large sky regions.

While any of the above methods could be used to provide depth data for our scene representation, we are primarily concerned with scene modelling, rather than dehazing, and therefore use the basic He et al. model [13].

4.2 Depth estimation

The typical objective of dehazing is to produce a clear image. Here we also want to obtain a depth map from the estimated transmission map. Atmospheric particles determine the *airlight* colour, which is effectively that of the haze, to mix with the scene radiance. This changes the observed colour distributions, and reduces the contrast of the images. The optical transmission map can be estimated, in the presence of haze, from a single image. The transmission map $t(x,y)$ is related to the range map $r(x,y)$, which encodes the distance to each visible scene point, via the Lambert-Beer law:

$$t(x,y) = \exp\big(-\gamma r(x,y)\big) \tag{17}$$

where γ is the haze density parameter. The radiance (haze free) image is estimated by standard methods [13], and stored in $\mathcal{L}_0^{\text{rgb}}(x,y)$, as described in Sec. 3.1. Each value in the range map

$$r(x,y) = -\log\big(t(x,y)\big)/\gamma \tag{18}$$

encodes the estimated radial distance from the camera centre to the scene point observed at $\mathcal{L}_0^{\text{rgb}}(x,y)$. The range map must be converted to a *depth* map, in which distances are measured parallel to the optical axis, for the model described in Sec. 3.2. Each pixel's depth is defined as

$$\mathcal{L}_0^z(x,y) = \cos(\varphi_{x,y})\, r(x,y) \tag{19}$$

$$\text{where} \quad \cos(\varphi_{x,y}) = \frac{f}{\sqrt{x^2+y^2+f^2}} \tag{20}$$

and f is the focal length of the camera. Note that the depth is less than the corresponding range, according to the cosine factor, except along the optical axis (where $z = r$). This transformation is performed before the segmentation and fitting procedures, which are described below.

4.3 Segmentation and fitting

To create a layered visualization, we *cluster* the radiance outputs $\mathcal{L}_0^{\text{rgb}}(x, y)$, and then fit the corresponding depths in $\mathcal{L}_0^{\text{z}}(x, y)$. To this end, we use an interactive graph-cut method [2]. The input into the clustering procedure is the radiance image $\mathcal{L}_0^{\text{rgb}}(x, y)$ and the output is an object mask $m_{i:N}$ for each cluster, where the number of clusters N is user defined. Users can interactively mark the foreground and background regions, using the mouse or touchpad. The graph-cut algorithm will then produce a binary mask image for the foreground region, based on the supplied constraints. We produce the foreground object and background image m_i', by mapping the mask image m_i and the input figure $\mathcal{L}_0^{\text{rgb}}(x, y)$. As the result may contain holes, the complete image w_i, we inpaint the holes (see Sec. 4.4) in the remaining background image m_i', which will be the input of the next iteration of clustering. Once all object masks $m_{1:N}$ are obtained, we produce a composed mask map $\mathcal{M}$, which has the same size as the original image, and assign a corresponding cluster label to each pixel.

We use RANSAC [7] to fit a depth plane to each cluster in $\mathcal{L}_0^{\text{z}}(x, y)$, as defined by the graph-cut procedure. RANSAC is reliable even in the presence of a high proportion of outliers, which is important here, because the dehazing sometimes produces bad depth estimates (e.g. when an object has the same colour as the airlight), and the clustering process may leave some pixels unassigned. The inputs into the plane fitting procedure are the mask map $\mathcal{M}$ and the depth map $\mathcal{L}_0^{\text{z}}(x, y)$ from equation (19). The output from plane fitting is a list of depth values of each plane vertices $\mathcal{L}_{1:N}^{\text{z}}$ for each cluster[2]. The orientations of different planes will vary, based on the distribution of points in each cluster. However, the crude estimation of depth maps from dehazing algorithms may result in inappropriate orientations for some planes. Hence, in our implementation, we chose to fix the planes in fronto-parallel orientations, as described in Sec. 5.1.

4.4 Inpainting

Image inpainting is the process of filling holes (areas that are occluded in an original view but could be visible in rendered virtual views [6]) based on the knowledge of neighbourhood regions. Criminisi et al. [3] introduced an exemplar-based method to fill holes with patches from surrounding regions. Their inpainting algorithm proceeds from the edge of holes to their centres, which helps maintain the linear structure in texture by combining colour and texture information. A challenge of image inpainting is to avoid propagating the wrong texture information from other objects. We will refer to this problem as 'inappropriate source'. To address this problem, depth constraints based on the exemplar inpainting can be introduced [5, 8]. These methods rely on the accuracy of depth information and, since the estimated depth maps from dehazing may contain errors (as in Fig. 15), we base the inpainting process on colour information.

Fig. 6 shows the pop-up visualization improvement achieved by applying the exemplar-based inpainting method, compared to [39]. Note that the holes caused by the foreground object are filled by semantic textures which fit well with the palace in the scene. However, the background layer keeps the inappropriate boundary of foreground objects. We refer to this problem as 'disrupted contour'.

(a) (b)

Fig. 6: Example of inpainting failure. (a): Before inpainting. (b): After direct exemplar-based inpainting [3], resulting in implausible content for the back layer [39]. The palace texture should not have been copied (red ellipse).

We will introduce two procedures based on Criminisi's algorithm [3], which address both the 'disrupted contour' and 'inappropriate source' issues, by alternating between clustering and inpainting, as illustrated in Fig. 7. An important preprocessing step is made to improve the quality of the inpainting procedure [3]. We observe that the image structure at a depth transition is often *not* representative of the occluded layer, because it will contain some mixture of the occluding layer (especially at complex boundaries, like foliage). Hence before proceeding with the inpainting process, the preprocessing is to perform an erosion of each cluster with an eight pixel radius after qualitatively comparison with other parameters.

To address the 'disrupted contour' problem, we alternate clustering and inpainting, as mentioned in Sec. 4.3. In the clustering procedure, users can see the texture and contours of each iteration's inpainting procedure. The semi-supervised graph-cut [26] clustering method allow users to segment the foreground objects with an appropriate boundary, which makes sense in the real world.

To address the 'inappropriate source' problem, we perform inpainting for all the holes caused by *each* previous clustering process. In clustering near to far objects, more and more pixels are segmented out from the inpainting source region. In particular, the source region of inpainting for the furthest layer will not contain any other objects, i.e. there will be no 'inappropriate source' problem. The process is illustrated in Fig. 8.

Another merit of our solution to 'inappropriate source' problem is that we avoid inpainted pixels to be a part of the source region. During the iterative procedure between clustering and inpainting, some inpainted pixels may remain after clustering as inputs into the following inpainting procedure. For the accuracy of inpainting, we do not want to propagate those estimated pixels into other target regions. We can solve this problem by re-inpainting on all previous holes. The size of holes increases along the alternating procedure between clustering and inpainting, which becomes computationally time consuming. To address this issue, we downsample the further layers adaptively. Since the further

2. This metadata is stored as a JSON file, which is read by the browser in our WEBGL implementation.

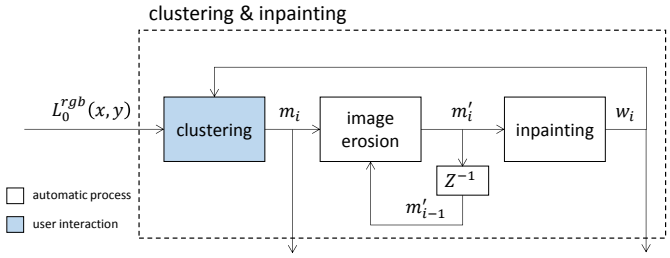


Fig. 7: Block diagram of the clustering & inpainting procedure. The loop from inpainting to clustering will run until all cluster masks $m_{1:N}$ are obtained. The total number of clusters N can be adjusted by users. Subscript i starts from 1 and increases by 1 up to N along the loop. Notice that the clustering procedure can be adapted to any automatic methods. A marker image m_i', composed of current cluster mask m_i and previous marker image m_{i-1}', is the input of the inpainting procedure after image erosion. The radiance image $\mathcal{L}_0^{\text{rgb}}(x,y)$ is labelled as m_0'. The output image w_i from inpainting will be the input to the next clustering stage.

layers often consist of relatively uniform background texture, and may be largely occluded by foreground objects, the downsampling has little effect on the final result.

4.5 Edge blending

The clustering process produces image segments with hard edges. These look very unconvincing in the pop-up visualisation, especially at complex depth boundaries (e.g. due to foliage). To make the transition more natural, we use the alpha channel $\mathcal{L}^\alpha$ in each foreground layer. We first compute the distance transform of each segment, with respect to any adjacent and *more distant* segments. This gives a scalar map $\delta(x,y)$, which encodes the image distance from pixel (x,y) to the nearest occluded point. The transition in the alpha channel $\alpha(x,y)$ is set based on the following rule:

$$\mathcal{L}^\alpha(x,y) = \begin{cases} \beta(\delta(x,y)/\epsilon) & \text{if } \delta(x,y) \le \epsilon \\ 1 & \text{otherwise} \end{cases} \quad (21)$$

where ϵ is the 'border width' of the depth transition (set to 5 pixels in all experiments here), and $\beta(\cdot)$ is a blending function. One possibility for the latter is simply $\beta(t) = t$, which results in a linear transition. However, the cubic Hermite function $\beta(t) = t^2(3-2t)$ is more suitable, since it has zero slope at both ends of the interval $t \in [0,1]$, which results in a smoother transition, as illustrated in Fig. 9. The outside edge pixels in the cubic edge blending result are softer, compared to the other results. We therefore apply cubic edge blending in our experiments.

5 Pop-up visualization

This section introduces the rendering method to visualize a visually pleasing pop-up model, with interactive viewpoint control. Five example pop-up scenes from different viewpoints and discussions on our model's limitations are provided. We also present visual comparisons with other single view 3D modelling methods.

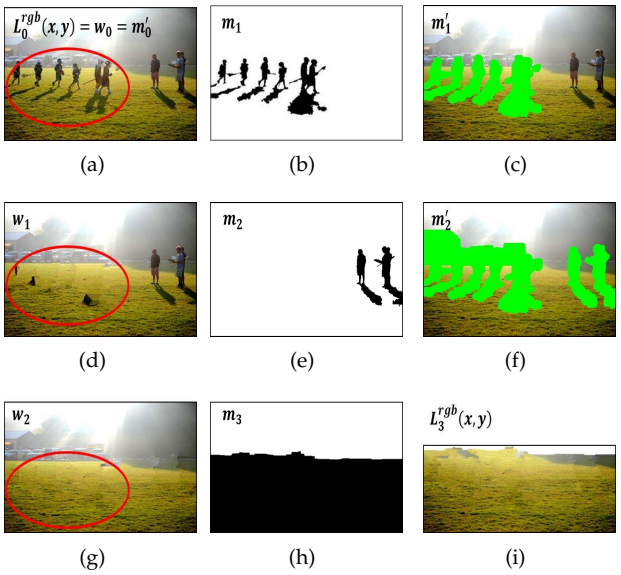


Fig. 8: Example of inpainting and clustering process. (a) The input radiance image. (b) The first mask image. (c) The marker image which is composed of (a) and (b). (d) The first inpainted image. (e) The second mask image is obtained with (d) as a new input into the clustering procedure. (f) The second marker image is composed of the current mask (e) and the previous marker image, as a new input into inpainting procedure. Note that we perform clustering and inpainting [3] alternately. In previous work [39], the layer texture was obtained from (d). In this work, the layer texture (i), is composed of the current mask image (h) and the previous inpainting result (g). The quality improvement of inpainting due to the removal of objects from the inpainting source region is highlighted with a red ellipses.

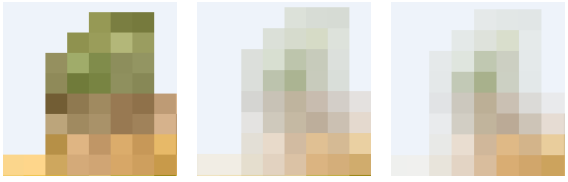
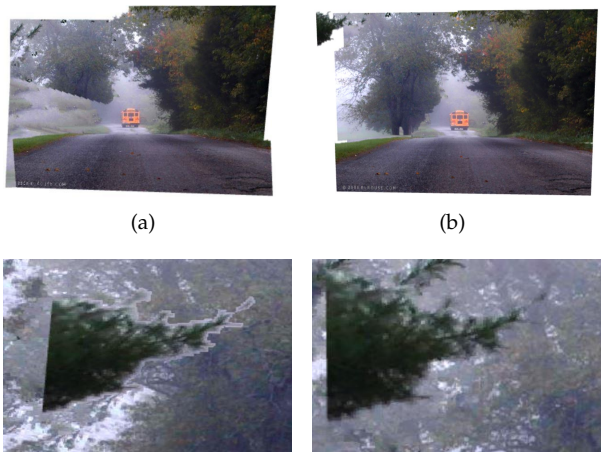


Fig. 9: Comparison between different edge blending methods in pixel detail corresponding to (a) solid, (b) linear blending and (c) cubic blending. The light blue background means full transparency.

5.1 Rendering

We visualize the pop-up scenes in a web browser, using the WebGL component of the Html5 standard. The final pop-up models can be rendered very efficiently, because WebGL automatically uses GPU shaders for depth testing and colour blending, where possible.

By default, WebGL will determine the visibility of each scene point, according to depth. Alternatively, we can define a fixed drawing order for the layers, to ensure the accuracy of the rendering process by avoiding any interference between depth and transparency, as shown in Fig. 10. It



(a) (b) (c) (d)

Fig. 10: Comparison of rendering results between previous and proposed methods. Image (a) is produced by our previous work [39], with intersection between two layers. Image (b) is the result in this work. The overlap issue is addressed. Image (c) shows the wrong border colours for the leaf, produced by our previous work [39]. Image (d) shows the correct result in this work.

may also be noted that this approach could facilitate an essentially 2D rendering approach, e.g. direct compositing in HTML5 Canvas. Depth order can be determined by sorting the depth centroids of the layers, so that the relationship of the textures can be recorded (e.g. in a corresponding JSON file). Ordering and depth testing are equivalent if the layers are non-intersecting, within their bounding boxes (see Fig. 11). A good compromise can be achieved by defining a drawing order that is sensitive to the distribution of opacities in each layer. Specifically, the opacity weighted centroid of layer i is

$$\overline{\mathbf{P}}_i = \frac{1}{A_i} \sum_{x,y} \mathcal{L}_i^{\alpha}(x,y) \left(\mathcal{R}(x,y) \cap \mathcal{L}_i\right) \quad \text{where} \quad A_i = \sum_{x,y} \mathcal{L}_i^{\alpha}(x,y) \tag{22}$$

is the total opacity of $\mathcal{L}_i$, and $\left(\mathcal{R}(x,y) \cap \mathcal{L}_i\right)$ is the geometric intersection of the ray through image point (x,y) with the layer. The drawing order is defined by sorting the depth coordinates z_i of the centroids, such that $\{z_{i_1}, \ldots, z_{i_N}\}$ is a non-decreasing sequence, as used in the composition (2). Note that $\overline{\mathbf{P}}_i \in \mathcal{L}_i$ geometrically, regardless of the layer orientation, by construction (22).

In our previous work [39], we used the image corners as the reference points for all quads. However, this can interfere with the rendering process, due to unnecessary depth testing of transparent surfaces. Hence we define 'tight' bounding rectangles, for each layer. We can determine the size of each box as follows:

$$x_{\min} = \min\{x : \mathcal{L}^{\alpha}(x,y) > 0\}$$
$$x_{\max} = \max\{x : \mathcal{L}^{\alpha}(x,y) > 0\}$$

and similarly for the y dimension, where the top-left of the rectangle is $(x_{\min}, y_{\min})$, and the bottom-right corner is $(x_{\max}, y_{\max})$. This scheme, in conjunction with the sorting procedure (22) gives good rendering performance. The four corners of each box are associated with 3D coordinates (X, Y, Z), which can be transformed as described below.

Fig. 11: Example of tight frames for each layer. The frames are highlighted by red lines, which contain the regions of nonzero opacity in the layer.

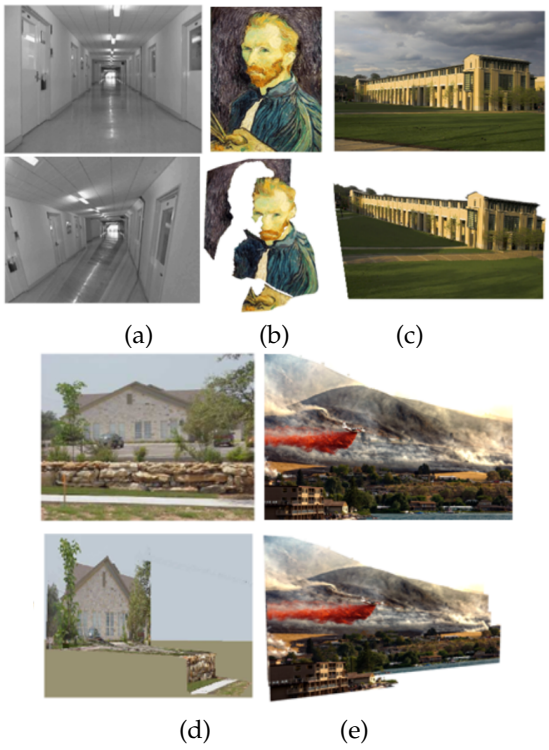
We use the THREEJS graphics library [1], which is built on WEBGL, to display the final models. Each layer is rendered as a quadrilateral, with the corresponding alpha premultiplied texture $\mathcal{L}^{\alpha}(x,y)$ obtained as described in Sec. 4. In our implementation, we allow users to manipulate pop-up scenes by holding and dragging the mouse (or the corresponding interaction tool in a touchpad).

5.2 Examples

We present five examples from different viewpoints in Fig. 12, which will be used in our subjective experiment introduced in Sec. 6. We select results from state-of-the-art scene rendering work of different types of model introduced in Fig. 3 to compare with our work. (Fig. 13). As an example, we apply our algorithm on an image from the Hoiem et al. [14] database. Note that the artefact in their result, due to overlapping vertical regions, is solved in our work (Fig. 14). By comparing with the other scene modelling methods qualitatively, the merits of our method are: 1) suitability for any scene geometry; 2) good performance at depth boundaries; 3) user controlled visualization system in web browser.

The limitations of our work are listed below. Since we use single hazy images, the geometry of each layer depends on the depth estimation results from the dehazing process. Dehazing algorithms can over-estimate the depth of white or grey objects, which may result in failures in the depth structure. Fig. 15 shows an example of a typical failure. The bench in the hazy scene has the similar colour with the grey airlight. The DCP dehazing algorithm [13] estimates the depth of grey objects by assuming those objects are a part of the haze. In this case, the bench object is assigned to the furthest layer, which is an error. However, any source for depth information obtained by hardware or image features can replace the dehazing procedure in our pipeline. In this case, our pipeline may also deal with indoor images with known range maps.

Fig. 12: Examples of pop-up scenes, generated with our method. Left column shows the original hazy input images. Middle and right columns show example novel views, taken from our pop-up models. From top to bottom are scenes from 1 to 5 in our experimental results, shown in Sec. 6.



(a) (b) (c) (d) (e)

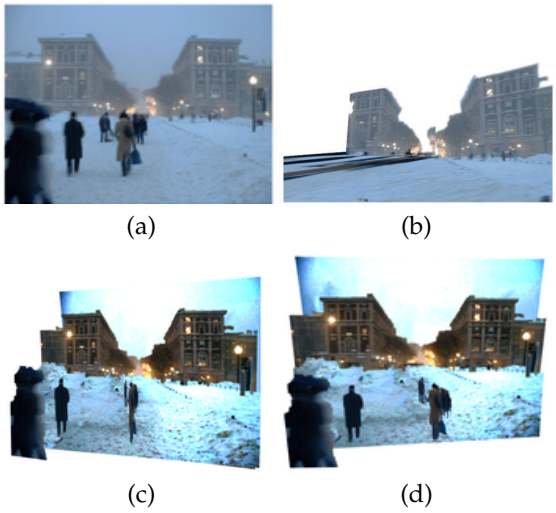
Fig. 13: Visual comparison with other single view 3D reconstruction methods. (a) Boxed model: Horry et al. [15]. (b) Meshed model with user guide: Zhang et al. [38]. (c)(d) Staggered model: Hoiem et al. [14] and Saxena et al. [28]. (e) Layered model: proposed method.

6 Perceptual Experiments

We investigate in two psychophysical experiments subjective preferences of depth structure. Both experiments were performed in a web browser, with user interactive mouse wheel control to explore the perceived depth distribution. During each experiment, the entire scene undergoes a slow automatic left-right change of viewpoint, in order to show the depth and occlusion structure. We asked the participants to interactively adjust the depth scale in order to achieve the most pleasing visual effect.[3].

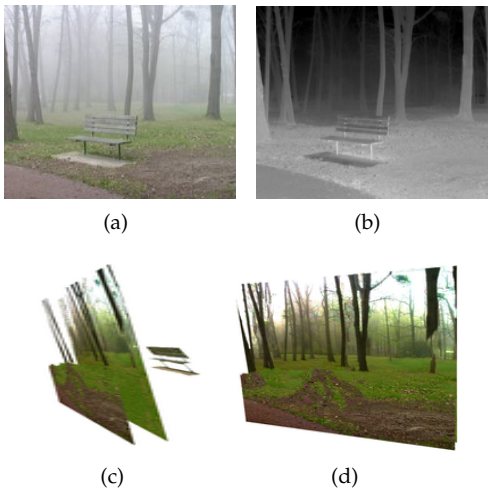
The first experiment investigates whether people have systematic preferences, for a given set of images, that can be used to resolve the scale ambiguity of the depths obtained by dehazing (because the haze coefficient is unknown) and of the other errors in the relative depths (e.g. due to inhomogeneous haze). In this first experiment, 10 viewers tested 5 high resolution pop-up scenes (shown in Fig. 12), which were initialised with three different depth structures. From top to bottom, we label the examples as scene 1 to scene 5 in the following discussion. The default depth structure scaling coefficient λ is 0.5 in equation (16). We have three initializations for each scene: 0.25, 0.5, and 0.75. The order of initialization is randomly set for each example. In order to decrease the experimental error, each case is presented three times [16]. Hence each viewer performed 45 trials in total (5 scenes $\times$ 3 initializations $\times$ 3 repetitions), in a random order.

3. See demo: http://cis.eecs.qmul.ac.uk/projects/popup/



(a) (b) (c) (d)

Fig. 14: Visual comparison with an experimental image taken from [14]. Original hazy image (a), and 3D model (b) by [14]. Note the streaking artefacts corresponding to the pedestrians in (b). Images (c) and (d) show other novel views generated from (a) using our system, with better depth and colour, in this case.



(a) (b) (c) (d)

Fig. 15: Example of depth failure. (a) The original image. (b) The depth map from dehazing. (c) A side view from our work. (d) A front view from our work. The colour of the bench is similar to that of the haze, leading to an overestimate of its depth (c). Nonetheless, the front view of the final model is visually consistent (d), although the bench has disappeared.

The second experiment is a further investigation, based on *synthetic* hazy images, for which the true depth structure is known. This experiment investigates the factors that influence the preferred depth structures. We repeat the first experiment using lidar depth data, and add synthetic haze to the input images. The layered model is created in exactly the same as for the first experiment, but in this case we can analyze it in relation to the ground-truth depth structure. Note that this test is relatively realistic, in that the depths are being estimated via the haze variations, as in the original experiment.

We use images and lidar range maps from the LIVE dataset [31, 32]. A weighted bilateral filter [30] is applied to fill the small gaps and holes in the range maps. We set the transmission of the sky to zero. Since the dataset was obtained in clear weather, we produce different densities synthetic uniform fog with various weather coefficients γ, and constant white airlight $\mathcal{A}$, according to Koschmieder's law [18]:

$$\begin{aligned} \mathcal{I}(x,y) &= t(x,y)\,\mathcal{J}(x,y) + \big(1 - t(x,y)\big)\mathcal{A} \\ t(x,y) &= \exp\big(-\gamma r(x,y)\big). \end{aligned} \tag{23}$$

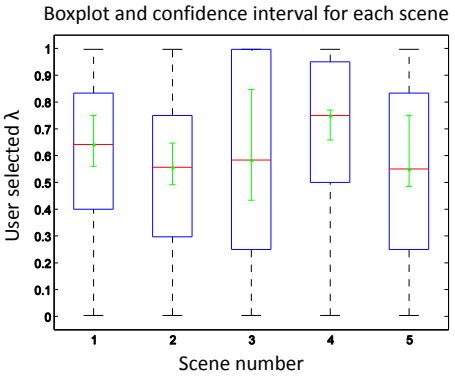
Here $\mathcal{I}$ is the synthetic hazy image, $\mathcal{J}$ is the approximate scene radiance, and $t(x,y)$ is the transmission map describing the portion of the scene light that reaches the camera. The transmission map is computed from the lidar range map $r(x,y)$, and chosen haze coefficient γ, as in equation (17). The latter is set to 0.005, 0.03, and 0.05, to provide a range of haze effects. We investigate the depth preferences for each haze level. Each of 20 participants completed 36 trials, with 4 scenes, 3 haze densities and 3 repetitions of each case.

We display the results based on each scene as box plots, shown in Fig. 16, and compute bootstrap estimates of the 95% confidence intervals of the group medians. The same analysis results on each participant in the first experiment are shown in Fig. 17. The same analysis results on each haze density in the second experiment are shown in Fig. 18. The subjective responses in Fig. 16 are highly variable, but with some scene dependent differences between the medians. The responses for scene 3 are particularly variable: this may be because the three foreground segmentations (trees, house and rainbow) are overlapping with each other, but at different depths. Because the inpainting process failed to provide a good texture in this case, users tried to conceal the errors by setting extreme depth structures (0 or 1). In general, the responses are somewhat concentrated in the upper-half of the range, which suggests that users liked to see a big depth difference between the closest and next closest layers. For example, in scene 4, the first layer is an individual trunk and the background layers containing the rest of the trees and the ground planes. A big depth variation between the first layer and the others will enhance the effect of 3D viewing. We can tell this from the small variation of the fourth confidence interval in the top box plot in Fig. 16.

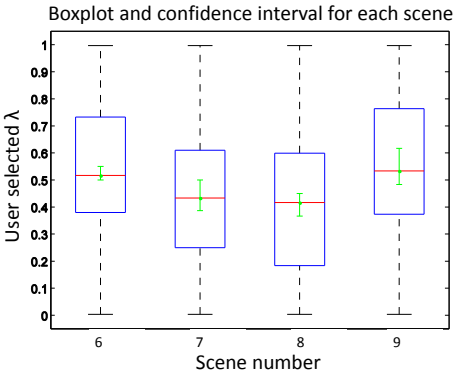
We make the following general observations. Firstly, the box plots in Fig. 16 (a) and Fig. 17 show highly variable choices of λ, which suggests that individual viewers do not have strong preferences (either according to scene, or in general). Furthermore, in some cases, users may have set the depth structure in order to conceal rendering artefacts, rather than to impose a general preference.

Despite this variability, some systematic differences between the median preferences are apparent, with respect to both scene and participant. The depth structure preferences from the experimental analysis are potentially helpful in obtaining estimated depth maps from the dehazing algorithm, to resolve the depth scaling ambiguity.

Fig. 18 shows no effect of the haze density: after collapsing over the different scenes, the data are indistinguishable. This suggests that enough depth structure was recovered in



(a)



(b)

Fig. 16: Depth preference by scene for all participants. 95% confidence intervals of the medians are shown in green. The responses are highly variable, but some systematic differences between the medians are apparent. (a) Box plot of five original hazy scenes in the first experiment. (b)-Top: Box plot of four ground-truth scenes with synthetic fog in the second experiment. (b)-Bottom: The scenes selected from the LIVE dataset [31, 32] numbered 6 to 9, from left to right.

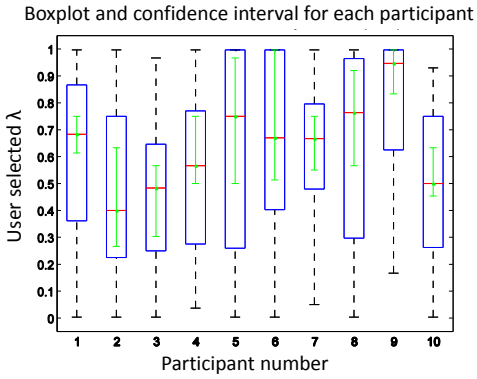


Fig. 17: Depth preferences by participant. Each box refers to a person's data on all scenes. The responses for each person are highly variable, but some individual differences are apparent.

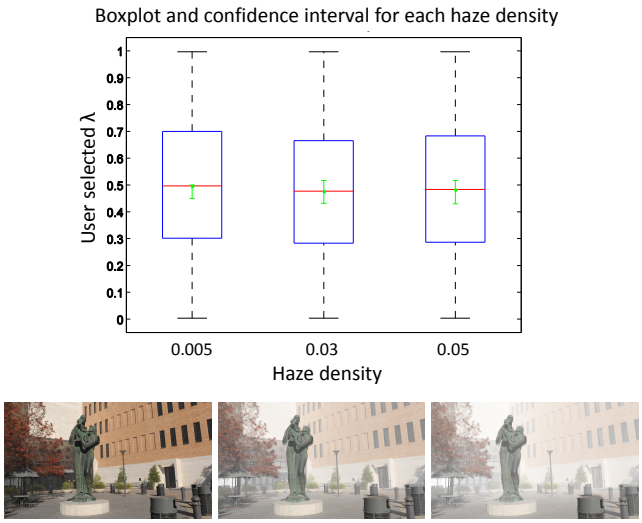


Fig. 18: Top: Box plot on each haze density with no effect by collapsing over all scenes. Bottom: Three examples of various haze densities: 0.005, 0.03, and 0.05 (from left to right), for scene 6.

all three cases, and that the final preferences were driven by the scene identity (including any rendering artefacts).

7 Conclusions

We have proposed a novel way to model 3D scenes, based on a single hazy image. The result can be viewed interactively in an ordinary web browser. We have shown that the problems of hole filling and edge blending can be addressed systematically, in relation to the layered depth structure. While the scope of this paper is limited to outdoor scenes in hazy conditions, the visualization pipeline could easily be adapted to any other source of depth information, including motion [37], or disparity [35]. Indeed, these cues could be suitably combined, according to the properties of the scene (e.g. estimated haze level).

In future work, to facilitate automatic single view 3D reconstruction, we plan to replace interactive graph-cut with an automatic method to segment the radiance image. This will additionally make use of the depth map, from the dehazing process, to perform RGBD clustering. It would also be interesting to study how the perceived quality of the final model depends on the number of clusters. In addition, ground-truth depth information could be used in a psychophysical study of depth structure preferences. Finally, we plan to explore semantic parsing of images. For example, we would like to apply the method introduced by Tighe et al. [36], which jointly infers scene labeling, object segmentation and relative depth ordering, by learning the relationships between occluders and background classes. We believe that these approaches, together with improved dehazing algorithms, will enable robust and automatic single view scene modelling.

Acknowledgments

The authors would like to thank all participants in the subjective evaluation experiment. The first author is grateful to the China Scholarship Council for financial support.

References

[1] Three JavaScript Library. http://threejs.org/.

[2] Boykov, Y. Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Computer Vision. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 105–112.

[3] Criminisi, A., Perez, P., and Toyama, K. (2003). Object removal by exemplar-based inpainting. In *Computer Vision and Pattern Recognition. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–II.

[4] Criminisi, A., Reid, I., and Zisserman, A. (2000). Single view metrology. *International Journal of Computer Vision*, 40(2):123–148.

[5] Daribo, I. and Pesquet-Popescu, B. (2010). Depth-aided image inpainting for novel view synthesis. In *Multimedia Signal Processing (MMSP), 2010 IEEE International Workshop on*, pages 167–170. IEEE.

[6] Fehn, C. (2004). Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3DTV. In *Electronic Imaging 2004*, pages 93–104. International Society for Optics and Photonics.

[7] Fischler, M. A. and Bolles, R. C. (1981). A paradigm for model fitting with applications to image analysis and automated cartography (reprinted in readings in computer vision, ed. MA Fischler). *Comm. ACM*, 24(6):381–395.

[8] Gautier, J., Le Meur, O., and Guillemot, C. (2011). Depth-based image completion for view synthesis. In *3DTV Conference: The True Vision-capture, Transmission and Display of 3D Video (3DTV-CON), 2011*, pages 1–4. IEEE.

[9] Gibson, K. B. and Nguyen, T. Q. (2013a). An analysis of single image defogging methods using a color ellipsoid framework. *EURASIP Journal on Image and Video Processing*, (1):37.

[10] Gibson, K. B. and Nguyen, T. Q. (2013b). Fast single image fog removal using the adaptive Wiener filter. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 714–718.

[11] Han, T. and Wan, Y. (2013). A fast dark channel prior-based depth map approximation method for dehazing single images. In *Information Science and Technology (ICIST), 2013 International Conference on*, pages 1355–1359. IEEE.

[12] Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.

[13] He, K., Sun, J., and Tang, X. (2011). Single image haze removal using dark channel prior. *IEEE transactions on pattern analysis and machine intelligence*, 33(12):2341–2353.

[14] Hoiem, D., Efros, A. A., and Hebert, M. (2005). Automatic photo pop-up. *ACM transactions on graphics (TOG)*, 24(3):577–584.

[15] Horry, Y., Anjyo, K.-I., and Arai, K. (1997). Tour into the picture: using a spidery mesh interface to make animation from a single image. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 225–232. ACM Press/Addison-Wesley Publishing Co.

[16] I.T.U.R. Assembly, U. (2003). *Methodology for the subjective assessment of the quality of television pictures*. International Telecommunication Union, Geneva, Switzerland.

[17] Karsch, K., Sunkavalli, K., Hadap, S., Carr, N., Jin, H., Fonte, R., Sittig, M., and Forsyth, D. (2014). Automatic scene inference for 3D object compositing. *ACM Transactions on Graphics (TOG)*, 33(3):32.

[18] Koschmieder, H. (1925). *Theorie der horizontalen sichtweite: kontrast und sichtweite*. Keim & Nemnich.

[19] Liebowitz, D., Criminisi, A., and Zisserman, A. (1999). Creating architectural models from images. In *Computer Graphics Forum*, volume 18, pages 39–50. Wiley Online Library.

[20] Matlin, E. and Milanfar, P. (2012). Removal of haze and noise from a single image. In *Computational Imaging*, page 82960T.

[21] Meng, G., Wang, Y., Duan, J., Xiang, S., and Pan, C. (2013). Efficient image dehazing with boundary constraint and contextual regularization. In *Proceedings of the IEEE international conference on computer vision*, pages 617–624.

[22] Narasimhan, S. G. and Nayar, S. K. (2000). Chromatic framework for vision in bad weather. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 598–605.

[23] Narasimhan, S. G. and Nayar, S. K. (2003). Contrast restoration of weather degraded images. *IEEE transactions on pattern analysis and machine intelligence*, 25(6):713–724.

[24] Nayar, S. K. and Narasimhan, S. G. (1999). Vision in bad weather. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 820–827.

[25] Porter, T. and Duff, T. (1984). Compositing digital images. In *ACM Siggraph Computer Graphics*, volume 18, pages 253–259.

[26] Rother, C., Kolmogorov, V., and Blake, A. (2004). Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23, pages 309–314.

[27] Saxena, A., Chung, S. H., and Ng, A. Y. (2005). Learning depth from single monocular images. In *NIPS*, volume 18, pages 1–8.

[28] Saxena, A., Chung, S. H., and Ng, A. Y. (2008). 3D depth reconstruction from a single still image. *International journal of computer vision*, 76(1):53–69.

[29] Saxena, A., Sun, M., and Ng, A. Y. (2007). Learning 3D scene structure from a single still image. In *Computer Vision. ICCV 2007. IEEE 11th International Conference on*, pages 1–8.

[30] Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pages 746–760. Springer.

[31] Su, C.-C., Bovik, A. C., and Cormack, L. K. (2011). Natural scene statistics of color and range. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 257–260.

[32] Su, C.-C., Cormack, L. K., and Bovik, A. C. (2013). Color and depth priors in natural images. *IEEE Transactions on Image Processing*, 22(6):2259–2274.

[33] Sutherland, I. E. (1974). Three-dimensional data input by tablet. *Proceedings of the IEEE*, 62(4):453–461.

[34] Świrski, L., Richardt, C., and Dodgson, N. A. (2011). Layered photo pop-up. In *ACM SIGGRAPH 2011 Posters*, page 36.

[35] Szeliski, R. and Golland, P. (1998). Stereo matching with transparency and matting. In *Computer Vision, 1998. Sixth International Conference on*, pages 517–524. IEEE.

[36] Tighe, J., Niethammer, M., and Lazebnik, S. (2014). Scene parsing with object instances and occlusion ordering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3748–3755.

[37] Wang, J. Y. and Adelson, E. H. (1994). Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638.

[38] Zhang, L., Dugas-Phocion, G., Samson, J.-S., and Seitz, S. M. (2002). Single-view modelling of free-form scenes. *Computer Animation and Virtual Worlds*, 13(4):225–235.

[39] Zhao, L., Hansard, M., and Cavallaro, A. (2015). Pop-up modelling of hazy scenes. In *International Conference on Image Analysis and Processing*, pages 306–318. Springer.

Lingyun Zhao received the B.S. degree from Beijing University of Posts and Telecommunication, China and Queen Mary, University of London, UK, in 2014. She is pursuing a Ph.D. degree in the Computer Vision Group, Queen Mary, University of London. Her current research includes 3D construction, image processing and computer vision.

Miles Hansard received the BSc, MRes, and the PhD degrees from University College London. He is a lecturer in computer science at Queen Mary, University of London. He is a member of the Vision Group, and of the QMUL Centre for Intelligent Sensing. His research interests include 3D scene modelling, depth cameras, and human vision.

Andrea Cavallaro received the Laurea (summa cum laude) from the University of Trieste, Italy, and the Ph.D. from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. Since 2003, he has been with Queen Mary University of London, UK, where he is Professor of Multimedia Signal Processing and Director of the Centre for Intelligent Sensing. His research interests include camera networks, target tracking, and perceptual semantics in multimedia.